# Refine Search

**Search Results -**

| Terms | Documents |
|---|---|
| L6 AND (symbol ADJ table) | 48 |

**Database:**
- US Pre-Grant Publication Full-Text Database
- US Patents Full-Text Database
- US OCR Full-Text Database
- EPO Abstracts Database
- JPO Abstracts Database
- Derwent World Patents Index
- IBM Technical Disclosure Bulletins

**Search:** L7

Refine Search

Recall Text | Clear | Interrupt

## Search History

**DATE: Tuesday, May 04, 2004** Printable Copy Create Case

| Set Name side by side | Query | Hit Count | Set Name result set |
|---|---|---|---|
| *DB=USPT; PLUR=NO; OP=OR* | | | |
| L7 | L6 AND (symbol ADJ table) | 48 | L7 |
| L6 | L5 AND debugger | 244 | L6 |
| L5 | L4 AND call | 745 | L5 |
| L4 | L2 AND Watchpoint OR breakpoint | 3227 | L4 |
| L3 | L2 AND Watchpoint OR breakpint | 10 | L3 |
| L2 | L1 AND symbol ADJ table | 446 | L2 |
| L1 | 714/37.ccls. OR 717/$$$.ccls. OR debug | 9146 | L1 |

END OF SEARCH HISTORY

# Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

**Search Results** - Record(s) 1 through 48 of 48 returned.

☑ 1. Document ID: US 6721941 B1

L7: Entry 1 of 48 File: USPT Apr 13, 2004

US-PAT-NO: 6721941
DOCUMENT-IDENTIFIER: US 6721941 B1

TITLE: Collection of timing and coverage data through a debugging interface

DATE-ISSUED: April 13, 2004

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Morshed; Farokh | Amherst | NH | | |
| Meagher; Robert | Milford | NH | | |

US-CL-CURRENT: 717/127; 709/217, 714/38, 717/124, 717/126, 717/128, 717/129, 717/130, 717/131, 719/328

ABSTRACT:

Techniques for gathering execution information about an application, such as a distributed application, are described. Key communication points in cross execution context calls, such as remote procedure calls, are determined and control is transferred to instrumentation routines to insert and extract execution information. Outgoing remote procedure calls are intercepted on a client that inserts call origin information into the request sent to a server system. Messages received by a server are intercepted. The server system extracts the call origin information and additionally inserts other information in a response sent to the client system upon completion of a remote procedure call. In turn, the client system intercepts the response and extracts other performance information. On each client and server system, information is gathered by a reader and forwarded to a local collector. This information may be further forwarded to and correlated by a client collector from one or more remote server collectors in accordance with processes of each distributed application. Various statistics for a distributed application may be determined in addition to per process statistics. These include wire time, code coverage as related to the distributed application, remote procedure call tracing, and performance profiling. A variety of techniques are described to obtain program execution information in connection with an executing application including instrumentation techniques and use of a debugger interface to obtain profiling and other execution information. All of the program execution data may be collected and correlated at one or more particular points using other techniques described to represent coordinated application monitoring.

50 Claims, 82 Drawing figures

Exemplary Claim Number: 1
Number of Drawing Sheets: 77

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw. D

---

☐ 2. Document ID: US 6704895 B1

L7: Entry 2 of 48 File: USPT Mar 9, 2004

US-PAT-NO: 6704895
DOCUMENT-IDENTIFIER: US 6704895 B1

TITLE: Integrated circuit with emulation register in JTAG JAP

DATE-ISSUED: March 9, 2004

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Daniels; Martin D. | Houston | TX | | |
| Coomes; Joseph A. | Missouri City | TX | | |

US-CL-CURRENT: 714/726; 714/30

ABSTRACT:

An emulation device including a serial scan testability interface having at least first and second scan paths, and state machine circuitry connected and responsive to said second scan path generally operable for emulation control.

3 Claims, 47 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 30

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw. D

---

☐ 3. Document ID: US 6691301 B2

L7: Entry 3 of 48 File: USPT Feb 10, 2004

US-PAT-NO: 6691301
DOCUMENT-IDENTIFIER: US 6691301 B2

TITLE: System, method and article of manufacture for signal constructs in a programming language capable of programming hardware architectures

DATE-ISSUED: February 10, 2004

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|

Bowen; Matt Littlemore GB

US-CL-CURRENT: 717/114; 712/15, 716/16

ABSTRACT:

A system, method and article of manufacture are provided for using a dynamic object in a programming language. In general, an object is defined with an associated first value and second value. The first value is used in association with the object during a predetermined clock cycle. The second value is used in association with the object before or after the predetermined clock cycle.

18 Claims, 129 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 117

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw De

☐ 4. Document ID: US 6546505 B1

L7: Entry 4 of 48 File: USPT Apr 8, 2003

US-PAT-NO: 6546505
DOCUMENT-IDENTIFIER: US 6546505 B1

TITLE: Processor condition sensing circuits, systems and methods

DATE-ISSUED: April 8, 2003

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Ehlig; Peter N. | Houston | TX | | |

US-CL-CURRENT: 714/30; 712/227

ABSTRACT:

A data processing device including a semiconductor chip, an electronic processor on-chip and an on-chip condition sensor connected to the electronic processor for analysis of the operations.

13 Claims, 93 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 60

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw De

☐ 5. Document ID: US 6539497 B2

L7: Entry 5 of 48 File: USPT Mar 25, 2003

US-PAT-NO: 6539497
DOCUMENT-IDENTIFIER: US 6539497 B2

TITLE: IC with selectively applied functional and test clocks

DATE-ISSUED: March 25, 2003

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Daniels; Martin D. | Houston | TX | | |

US-CL-CURRENT: 714/30; 713/601, 714/726, 714/727

ABSTRACT:

A data processing device formed in a single semiconductor chip. The data processing device includes an electronic processor, and on-chip peripheral circuitry ordinarily operative together. Further included, are means for selectively entering externally supplied data into the electronic processor and on-chip peripheral circuitry, for starting and stopping operations of the electronic processor and the on-chip peripheral circuitry independently of each other in an emulation mode of operation.

5 Claims, 47 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 30



☐ 6. Document ID: US 6530079 B1

L7: Entry 6 of 48 File: USPT Mar 4, 2003

US-PAT-NO: 6530079
DOCUMENT-IDENTIFIER: US 6530079 B1

TITLE: Method for optimizing locks in computer programs

DATE-ISSUED: March 4, 2003

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Choi; Jong-Deok | Mount Kisco | NY | | |
| Gupta; Manish | Peekskill | NY | | |
| Serrano; Mauricio J. | San Jose | CA | | |
| Sreedhar; Vugranam C. | Whiteplains | NY | | |
| Midkiff; Samuel Pratt | Upper Saddle River | NJ | | |

US-CL-CURRENT: 717/158; 712/227, 717/127, 717/128, 717/129, 717/130, 717/131, 717/152, 717/153, 717/157, 717/159

ABSTRACT:

A method and several variants for using information about the scope of access of objects acted upon by mutual exclusion, or mutex, locks to transform a computer program by eliminating locking operations from the program or simplifying the locking operations, while strictly performing the semantics of the original program. In particular, if it can be determined by a compiler that the object locked can only be accessed by a single thread it is not necessary to perform the "acquire" or "release" part of the locking operation, and only its side effects must be performed. Likewise, if it can be determined that the side effects of a locking operation acting on a variable which is locked in multiple threads are not needed, then only the locking operation, and not the side effects, needs to be performed. This simplifies the locking operation, and leads to faster programs which use fewer computer processor resources to execute; and programs which perform fewer shared memory accesses, which in turn not only causes the optimized program, but also other programs executing on the same computing machine to execute faster. The method also describes how information about the semantics of the locking operation side effects and the information about the scope of access can also be used to eliminate performing the side effect parts of the locking operation, thereby completely eliminating the locking operation. The method also describes how to analyze the program to compute the necessary information about the scope of access. Variants of the method show how one or several of the features of the method may be performed.

21 Claims, 2 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 2



---

☐ 7. Document ID: US 6522985 B1

L7: Entry 7 of 48 File: USPT Feb 18, 2003

US-PAT-NO: 6522985
DOCUMENT-IDENTIFIER: US 6522985 B1

TITLE: Emulation devices, systems and methods utilizing state machines

DATE-ISSUED: February 18, 2003

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Daniels; Martin D. | Houston | TX | | |
| Coomes; Joseph A. | Missouri City | TX | | |

US-CL-CURRENT: 702/117; 702/108, 702/118, 714/726, 714/729, 714/733

ABSTRACT:

An emulation device including a serial scan testability interface having at least first and second scan paths, and state machine circuitry connected and responsive to said second scan path generally operable for emulation control of logical circuitry associated with said emulation device.

21 Claims, 47 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 30



---

☑ 8. Document ID: US 6425119 B1

L7: Entry 8 of 48 File: USPT Jul 23, 2002

US-PAT-NO: 6425119
DOCUMENT-IDENTIFIER: US 6425119 B1

TITLE: Method to produce application oriented languages

DATE-ISSUED: July 23, 2002

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Jones; Mark A. | New Providence | NJ | | |
| Nakatani; Lloyd H. | Westfield | NJ | | |

US-CL-CURRENT: 717/100; 717/114

ABSTRACT:

Jargons are a family of application oriented languages well-suited for representing and processing complex, hierarchically structured information. A system is presented that automates most of the work of making a jargon, so practically any programmer can make a simple one in a few days. Every jargon has the same syntax, is processed with same ready-made base interpreter, and comes complete with a suite of "deluxe" features: debugger, error handler, function definition, associative arrays, variables, incremental loader, among others. The system provides a general purpose programming language for writing actions that defines the semantics of a jargon and an interpreter written in the general purpose language and customized for the jargon, by integrating the jargon's actions into the interpreter. Using jargons, the same information document may be reprocessed to generate a multiplicity of products.

16 Claims, 5 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 2



---

☐ 9. Document ID: US 6412106 B1

L7: Entry 9 of 48 File: USPT Jun 25, 2002

US-PAT-NO: 6412106
DOCUMENT-IDENTIFIER: US 6412106 B1

TITLE: Graphical system and method for debugging computer programs

DATE-ISSUED: June 25, 2002

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Leask; Gary M. | Dallas | TX | | |
| Huffman; Dale L. | Allen | TX | | |

US-CL-CURRENT: 717/124; 717/125, 717/148

ABSTRACT:

A system and method for graphically debugging a computer program is disclosed. In a preferred embodiment, a graphical debugging environment is provided, which is capable of displaying a graphical representation of an application program to be debugged. Thereafter, the graphical debugging environment allows a user to insert debugging tools, such as breakpoints, directly into the graphical representation of the application program. Thus, a user is not required to interact with the textual source code of an application program when debugging it. The graphical debugging environment may display indicators illustrating where debug tools have been inserted within the application program. In a preferred embodiment, the graphical debugging environment allows a user to perform debugging during an application program's runtime. Thus, a user is not required to halt an application program prior to debugging it. Also, in a preferred embodiment the graphical debugging environment executing on a local computer may be used to debug an application program residing on a remote computer.

52 Claims, 9 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 7



---

☐ 10. Document ID: US 6374369 B1

L7: Entry 10 of 48 File: USPT Apr 16, 2002

US-PAT-NO: 6374369
DOCUMENT-IDENTIFIER: US 6374369 B1

TITLE: Stochastic performance analysis method and apparatus therefor

DATE-ISSUED: April 16, 2002

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| O'Donnell; Ciaran | Santa Clara | CA | | |

US-CL-CURRENT: 714/38; 717/130, 717/153

ABSTRACT:

A programmable method for analyzing the performance of software using a combination of statistical sampling, hardware events and feedback, and a finite state machine execution model. Performance analysis code is integrated with the object code of the software it is analyzing and profiling. Using hardware timers and triggers, the analysis code records timing information at each timer or trigger event, where some events may be the result of stochastic sampling. At certain times during the execution and termination of the software being profiled, results of the profiling are output. Upon the termination of the software being profiled, post processing is optionally performed on the profiling output timing information, and the result of this post processing provides a human readable indication of where the analyzed software spent its execution time. A system for implementing the profiling method in hardware is also described.

46 Claims, 16 Drawing figures
Exemplary Claim Number: 17
Number of Drawing Sheets: 8



---

☐ 11. Document ID: US 6349392 B1

L7: Entry 11 of 48 File: USPT Feb 19, 2002

US-PAT-NO: 6349392
DOCUMENT-IDENTIFIER: US 6349392 B1

TITLE: Devices, systems and methods for mode driven stops

DATE-ISSUED: February 19, 2002

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Daniels; Martin D. | Houston | TX | | |

US-CL-CURRENT: 714/30; 713/601, 714/727

ABSTRACT:

A data processing device formed in a single semiconductor chip. The data processing device includes an electronic processor, and on-chip peripheral circuitry ordinarily operative together. Further included, are means for selectively entering externally supplied data into the electronic processor and on-chip peripheral circuitry, for starting and stopping operations of the electronic processor and the on-chip peripheral circuitry independently of each other in an emulation mode of operation.

1 Claims, 47 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 30

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw D

---

☐ 12. Document ID: US 6324683 B1

L7: Entry 12 of 48 File: USPT Nov 27, 2001

US-PAT-NO: 6324683
DOCUMENT-IDENTIFIER: US 6324683 B1
**** See image for Certificate of Correction ****

TITLE: System, method and program for debugging external programs in client/server-based relational database management systems

DATE-ISSUED: November 27, 2001

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Fuh; You-Chin Gene | San Jose | CA | | |
| Meier; Michael Stephen | Newark | CA | | |
| Pan; Hsin | San Jose | CA | | |

US-CL-CURRENT: 717/124

ABSTRACT:

The present invention provides a method, system, and program for debugging external programs, such as user-defined functions, stored procedures, and triggers executed in relational database management systems (RDBMS), in a client-server, i.e., distributed, environment. In the present invention, a debugger is initiated from within a process running the external program by executing a special segment of code prior to the execution of the external program. In one embodiment of the invention, this debugging function is triggered by including a variation of this special segment of program code within the external program, itself. In another embodiment of the invention, this debugging triggering function is provided within an enhanced RDBMS with extensions to SQL to activate the debugging ability in the RDBMS. The invention can be implemented by using present day serial debuggers or parallel and/or distributed debuggers. One such parallel and distributed debugger utilized in a preferred embodiment is the Parallel and Distributed Dynamic Analyzer (PDDA) debugger. In addition, although the following invention is described with reference to a debugger, the invention can be applicable for any application development tool.

19 Claims, 17 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 16

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw D

☑ 13. Document ID: US 6275956 B1

L7: Entry 13 of 48 File: USPT Aug 14, 2001

US-PAT-NO: 6275956
DOCUMENT-IDENTIFIER: US 6275956 B1

TITLE: Integrated dynamic-visual parallel debugging apparatus and method thereof

DATE-ISSUED: August 14, 2001

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| On; Gi Won | Daejeon | | | KR |
| Lee; Bum Sik | Daejeon | | | KR |
| Chi; Dong Hae | Daejeon | | | KR |
| Park; Chee Hang | Daejeon | | | KR |

US-CL-CURRENT: 717/125; 714/38, 717/128

ABSTRACT:

An integrated dynamic-visual parallel debugging apparatus comprising: replay drive means for receiving a program and creating program symbol table information and an execution log file through reference execution; a parallel debugger core for receiving the program symbol table information and the execution log file from the replay drive means and managing views and events; and graphical user interface means for interfacing the parallel debugger core and a user.

15 Claims, 6 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 6



☑ 14. Document ID: US 6249907 B1

L7: Entry 14 of 48 File: USPT Jun 19, 2001

US-PAT-NO: 6249907
DOCUMENT-IDENTIFIER: US 6249907 B1

TITLE: Method system and article of manufacture for debugging a computer program by encoding user specified breakpoint types at multiple locations in the computer program

DATE-ISSUED: June 19, 2001

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Carter; Derek Kneil | Morgan Hill | CA | | |

| | | | |
|---|---|---|---|
| Wessels; Ronald | Mississauga | | CA |
| Yukihiro; Della Ann | San Jose | CA | |

US-CL-CURRENT: 717/129; 717/130

ABSTRACT:

Disclosed is a system for debugging a computer program. A user indicates a specified breakpoint type, such as a program statement, variable reference, command, etc. The program, including program statements, is then compiled. During compilation, the compiler locates statements in the program corresponding to the breakpoint types and generates a function call into the program at instances in the program of statements corresponding to the user specified breakpoint types. During a debugging phase, a debugger may execute an executable version of the program, including the function calls. Upon processing the function calls, the debugger may stop execution of the program and pass control to the user to perform debugging operations.

20 Claims, 5 Drawing figures
Exemplary Claim Number: 4
Number of Drawing Sheets: 4



☐ 15. Document ID: US 6219782 B1

L7: Entry 15 of 48 File: USPT Apr 17, 2001

US-PAT-NO: 6219782
DOCUMENT-IDENTIFIER: US 6219782 B1
**** See image for Certificate of Correction ****

TITLE: Multiple user software debugging system

DATE-ISSUED: April 17, 2001

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Khan; Azeemullah | Redmond | WA | | |
| Noyama; Glenn T. | Kirkland | WA | | |
| Pennell; Andrew Michael | Redmond | WA | | |

US-CL-CURRENT: 712/227; 709/203

ABSTRACT:

A minimally intrusive debugging system for use by multiple users for concurrently and independently debugging a common software target in a client and server debugging environment. The target software is a non-compiled interpreted script-type program that is individually controlled by independent client debugging sessions. Each debug engine in the client's debugging session is used to control the target software program using debug system library interface calls that are integrated into the executing target software program. The debug system library

interface calls facilitate communication of target system program events to the client's debug engine and to extract internal operational information from said target software program by the client debug engine and target software program interface on each client computing device.

24 Claims, 6 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 5

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw De

---

☑ 16. Document ID: US 6161216 A

L7: Entry 16 of 48 File: USPT Dec 12, 2000

US-PAT-NO: 6161216
DOCUMENT-IDENTIFIER: US 6161216 A

TITLE: Source code debugging tool

DATE-ISSUED: December 12, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Shagam; Eli | Brookline | MA | | |

US-CL-CURRENT: 717/128; 717/132, 717/140

ABSTRACT:

A method and apparatus for debugging the source code using the source code debugger includes the following steps. A script generator is provided to receive source code instructions. Executing the script generator includes reading each source code instruction and generating, based on the type of instruction, a debugging script. The debugging script includes a specification of trace points. The debugging script is then provided to the source code debugger.

18 Claims, 8 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 8

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw De

---

☐ 17. Document ID: US 6091896 A

L7: Entry 17 of 48 File: USPT Jul 18, 2000

US-PAT-NO: 6091896
DOCUMENT-IDENTIFIER: US 6091896 A

TITLE: Debugging optimized code using data change points

DATE-ISSUED: July 18, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Curreri; Donald L. | Woburn | MA | | |
| Iyengar; Arun K. | Yorktown Heights | NY | | |
| Biesele; Russell A. | Santa Clara | CA | | |
| Ruscetta; Michelle A. | San Jose | CA | | |

US-CL-CURRENT: 717/125; 717/129

ABSTRACT:

A software development system with improved facilities for debugging optimized code. Certain parts of the operations specified by source statements are categorized as "data change points". A compiler 102 identifies the correspondence between the machine instructions that perform data change point operations and the source statements from which these machine instructions were derived and stores this mapping data 110 for use by a debugger program 104. This source code/machine instruction mapping data is used by the debugger to permit users to specify machine instruction locations (such as for setting breakpoints) by identifying a source statement and specifying that the debugger use the data change point of that statement; as another example, the step command includes an option that permits a user to continue execution until the next data change point instruction is reached. The debugger's source code display (FIGS. 5A and 5B) is also adapted to take advantage of the data change point mapping data: a user can request that the debugger highlight source statements for which the data change point instruction has executed; in marking the source display to indicate the location of the current execution point, the debugger can also show whether the next instruction to execute will be a data change point instruction.

20 Claims, 7 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 2



---

☐ 18. Document ID: US 6085336 A

L7: Entry 18 of 48 File: USPT Jul 4, 2000

US-PAT-NO: 6085336
DOCUMENT-IDENTIFIER: US 6085336 A

TITLE: Data processing devices, systems and methods with mode driven stops

DATE-ISSUED: July 4, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Daniels; Martin D. | Houston | TX | | |

US-CL-CURRENT: 714/30; 714/727

ABSTRACT:

A data processing device formed in a single semiconductor chip. The data processing device includes an electronic processor, and on-chip peripheral circuitry ordinarily operative together. Further included, are means for selectively entering externally supplied data into the electronic processor and on-chip peripheral circuitry, for starting and stopping operations of the electronic processor and the on-chip peripheral circuitry independently of each other in an emulation mode of operation.

32 Claims, 46 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 53



---

☑ 19. Document ID: US 6067641 A

L7: Entry 19 of 48 File: USPT May 23, 2000

US-PAT-NO: 6067641
DOCUMENT-IDENTIFIER: US 6067641 A

TITLE: Demand-based generation of symbolic information

DATE-ISSUED: May 23, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| McInerney; Peter J. | Cupertino | CA | | |
| You; Lawrence L. | San Jose | CA | | |
| Wimble; Michael D. | Sunnyvale | CA | | |

US-CL-CURRENT: 714/38; 712/227

ABSTRACT:

A human-oriented object programming system (HOOPS) and its debugger provide an interactive and dynamic modeling system to assist in the incremental generation of symbolic information of computer programs that facilitates the development of complex computer programs such as operating systems and large applications with graphic user interfaces (GUIs). A program is modeled as a collection of units called components. A component represents a single compilable language element such as a class or a function. One major functionality built in HOOPS is the debugger, using symbolic properties. The database stores the components and properties. The debugger, using a GUI, displays to the user the execution state of the program. To display the execution state in terms of the programmer's source code, the debugger demands retrieval and/or generation of the symbolic properties of the program. The compiler, which is responsible for calculating the dependencies associated with a component, uses those dependencies to generate the information stored in symbolic properties. The debugger matches versions of source and object code and retrieves source code configuration as needed. Symbolic properties that are stored in the

database can be removed to reduce database and disk memory usage; they can be later reconstructed using the same method of demand-based generation of symbolic information.

8 Claims, 43 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 40



---

☐ 20. Document ID: US 6058393 A

L7: Entry 20 of 48 File: USPT May 2, 2000

US-PAT-NO: 6058393
DOCUMENT-IDENTIFIER: US 6058393 A

TITLE: Dynamic connection to a remote tool in a distributed processing system environment used for debugging

DATE-ISSUED: May 2, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Meier; Michael Stephen | Newark | CA | | |
| Pan; Hsin | San Jose | CA | | |

US-CL-CURRENT: 707/10; 714/38, 714/45, 717/124, 718/100

ABSTRACT:

The present invention provides a dynamic connection for distributed applications that need to locate application development tools, including but not limited to debuggers, trace collection tools, compilers, etc.) which may be running on different machines, and to send the tools messages. The program requesting debugging service (i.e., a debugger client) sends, to a tool locator, criteria which specifies the properties of a desired debugger. The tool locator maintains a registry of all tools, e.g. debuggers, and their properties, which remain active within the network by receiving tool registration information from each tool as it is started on any machine within the network. When a message is received by the tool locator from a debugger client specifying the criteria of a desired debugger, the tool locator searches its registry and returns a list of debuggers matching the specified properties along with a communication endpoint address that can be used to establish a connection with a debugger meeting the criteria. The debugger client then sends a message, using the established connection, to the desired debugger

requesting debugging services on behalf of the debugger client or another program. As a result, a dynamic connection is made, at run time, between an application program and a debugger having certain desired properties wherein the debugger may be active, if at all, at any time on any machine within the network.

30 Claims, 9 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 9

☐ 21. Document ID: US 6041176 A

L7: Entry 21 of 48 File: USPT Mar 21, 2000

US-PAT-NO: 6041176
DOCUMENT-IDENTIFIER: US 6041176 A

TITLE: Emulation devices utilizing state machines

DATE-ISSUED: March 21, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Shiell; Jonathan H. | Plano | TX | | |

US-CL-CURRENT: 703/27; 712/210

ABSTRACT:

An emulation device which enables a functional circuit to support self emulation. A serial scan testability interface has at least first, second and third scan paths, said first scan path being provided for applying digital information to the functional circuit for use in emulation of the functional circuit. A first state machine connected to said second scan path has a first state selected from among a first set of states. A second state machine connected to said third scan path has a second state selected from among a second set of states. The emulation device performs an emulation command based on a combined first state of said first state machine and second state of said second state machine. The state of the first state machine indicates a primary portion of the emulation command denoting an emulation command class. The state of the second state machine indicates a secondary portion of the emulation command consisting of a subtype within the emulation command class. Both state machines return to a default "no-op" state following performance of an emulation command prior to accepting new data. The first scan path is used for JTAG when the first state machine is the default "no-op" state. Alternatively, the first scan path is used for JTAG when the first state machine is a "normal JTAG operation" command class. An unused JTAG command indicates to the first state machine that the following bit string is a command class and not a JTAG operation.

32 Claims, 26 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 18

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw De

☐ 22. Document ID: US 6032268 A

L7: Entry 22 of 48 File: USPT Feb 29, 2000

US-PAT-NO: 6032268
DOCUMENT-IDENTIFIER: US 6032268 A

TITLE: Processor condition sensing circuits, systems and methods

DATE-ISSUED: February 29, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Ehlig; Peter N. | Houston | TX | | |

US-CL-CURRENT: 714/30; 714/34, 714/726

ABSTRACT:

The invention provides improved architectures and methods for emulation, simulation, and testability of data processing devices and systems without requiring physical probing or special test fixtures. A data processing device may include a semiconductor chip that is divided into domains. One domain may be halted and tested while another domain continues to operate. For example, the semiconductor chip may have a electronic processor domain and an analysis domain. The analysis domain may include an on-chip condition sensor that is connected to the electronic processor domain. The chip can further include control logic circuitry to allow the analysis domain to operate while the electronic processor is halted and tested.

18 Claims, 47 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 41


---

☐ 23. Document ID: US 6026235 A

L7: Entry 23 of 48 File: USPT Feb 15, 2000

US-PAT-NO: 6026235
DOCUMENT-IDENTIFIER: US 6026235 A

TITLE: System and methods for monitoring functions in natively compiled software programs

DATE-ISSUED: February 15, 2000

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Shaughnessy; Steven T. | Scotts Valley | CA | | |

US-CL-CURRENT: 717/127; 717/130

ABSTRACT:

A development system having a monitor/profiler tool for monitoring functions in natively compiled software programs is described. According to the present invention, the monitor/profiler tool is constructed to work directly on a natively compiled software application which only have debugging info. Unlike prior approaches, the monitor/profiler tool does not require a special compile or link phase for the application under exam. The tool can monitor any function in software application which has debug info, thus relieving program developers from the burden of maintaining two ways of building an application. The developer can simply use the same executable for both development and function analysis.

12 Claims, 7 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 6



---

☐ 24. Document ID: US 5983366 A

L7: Entry 24 of 48 File: USPT Nov 9, 1999

US-PAT-NO: 5983366
DOCUMENT-IDENTIFIER: US 5983366 A

TITLE: Data processing system having monitoring of software activity

DATE-ISSUED: November 9, 1999

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| King; Michael Roy | Axams | | | AT |

US-CL-CURRENT: 714/38

ABSTRACT:

A method of tracing execution of a computer program of a data processing system with print messages is disclosed. The method includes providing a trace definition file containing a plurality of trace definitions, each trace definition is an encoded identification of each print message to be used with the tracing of execution of the computer program of the data processing system and a format display specification of an output for providing a trace of the execution of the computer program; providing information from the trace definition file to the data processing system which is executed with the computer program; providing information from the trace definition file to a host processor; the host processor communicating with the data processing system to identify selected parts of the computer program to be traced; in response to execution of the computer program transmitting trace information from the data processor to the host processor associated with the selected parts of the application program being traced; and the host processor, in response to transmission of the trace information from the data processor and in response to the information provided from the trace definition file causing the output for enabling tracing of the execution of the selected parts of the computer program.

36 Claims, 5 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 5

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Drawi De

☐ 25. Document ID: US 5956512 A

L7: Entry 25 of 48 File: USPT Sep 21, 1999

US-PAT-NO: 5956512
DOCUMENT-IDENTIFIER: US 5956512 A
**** See image for Certificate of Correction ****

TITLE: Computer program debugging in the presence of compiler synthesized variables

DATE-ISSUED: September 21, 1999

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Simmons; Steven M. | Dallas | TX | | |
| Brooks; Gary S. | Garland | TX | | |

US-CL-CURRENT: 717/128; 717/154, 717/162

ABSTRACT:

A debugger is used in an environment of optimized compiling to track both user-defined and synthesized variables so that the values of these variables at selected programmer counter addresses can be either determined or set. The tracking is primarily accomplished by the generation of various interrelated tables including a Type Scope Table, a Name Space Table, an Expression Table, a Location Range Tab and a Variable Table. These tables define the existence of variable at defined program counter ranges and provide the algebraic definitions for the synthesized variables. A programmer can efficiently debug a program produced with optimized compiling through the operations of determining variable values and setting variable values.

8 Claims, 15 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 9

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Drawi De

☑ 26. Document ID: US 5956479 A

L7: Entry 26 of 48 File: USPT Sep 21, 1999

US-PAT-NO: 5956479
DOCUMENT-IDENTIFIER: US 5956479 A

TITLE: Demand based generation of symbolic information

DATE-ISSUED: September 21, 1999

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| McInerney; Peter J. | Cupertino | CA | | |
| You; Lawrence L. | San Jose | CA | | |
| Wimble; Michael D. | Sunnyvale | CA | | |

US-CL-CURRENT: 714/38; 715/532, 717/127

ABSTRACT:

A human oriented object programming system (HOOPS) and its debugger provide an interactive and dynamic modeling system to assist in the incremental generation of symbolic information of computer programs which facilitates the development of complex computer programs such as operating systems and large applications with graphic user interfaces (GUIs). A program is modeled as a collection of units called components. A component represents a single compilable language element such as a class or a function. One major functionality built on HOOPS is the debugger, using symbolic properties. The database stores the components and properties. The debugger, using a GUI, displays to the user the execution state of the program. To display the execution state in terms of the programmer's source code, the debugger demands retrieval and/or generation of the symbolic properties of the program. The compiler, which is responsible for calculating the dependencies associated with a component, uses those dependencies to generate the information stored in symbolic properties. The debugger matches versions of source and object code and retrieves source code configuration as needed. Symbolic properties that are stored in the database can be removed to reduce database and disk memory usage; they can be later reconstructed using the same method of demand-based generation of symbolic information.

24 Claims, 43 Drawing figures
Exemplary Claim Number: 9
Number of Drawing Sheets: 40



---

☐ 27. Document ID: US 5892941 A

L7: Entry 27 of 48 File: USPT Apr 6, 1999

US-PAT-NO: 5892941
DOCUMENT-IDENTIFIER: US 5892941 A

TITLE: Multiple user software debugging system

DATE-ISSUED: April 6, 1999

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Khan; Azeemullah | Redmond | WA | | |
| Noyama; Glenn T. | Kirkland | WA | | |

Pennell; Andrew Michael Redmond WA

US-CL-CURRENT: 703/22

ABSTRACT:

A minimally intrusive debugging system for use by multiple users for concurrently and independently debugging a common software target in a client and server debugging environment. The target software is a non-compiled interpreted script-type program that is individually controlled by independent client debugging sessions. Each debug engine in the client's debugging session is used to control the target software program using debug system library interface calls that are integrated into the executing target software program. The debug system library interface calls facilitate communication of target system program events to the client's debug engine and to extract internal operational information from said target software program by the client debug engine and target software program interface on each client computing device.

18 Claims, 6 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 5



---

☐ 28. Document ID: US 5841670 A

L7: Entry 28 of 48 File: USPT Nov 24, 1998

US-PAT-NO: 5841670
DOCUMENT-IDENTIFIER: US 5841670 A

TITLE: Emulation devices, systems and methods with distributed control of clock domains

DATE-ISSUED: November 24, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |

US-CL-CURRENT: 703/23; 714/731

ABSTRACT:

An emulation device (11) distributes common control information (8801) to each of a plurality of clock domains (1213, 1215, 1217) into which the emulation device is partitioned, and also provides the clock domains with individualized clock control (8905, 8907, 8913).

15 Claims, 94 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 50

☐ 29. Document ID: US 5809283 A

L7: Entry 29 of 48 File: USPT Sep 15, 1998

US-PAT-NO: 5809283
DOCUMENT-IDENTIFIER: US 5809283 A
**** See image for Certificate of Correction ****

TITLE: Simulator for simulating systems including mixed triggers

DATE-ISSUED: September 15, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Vaidyanathan; Radha | Los Altos | CA | | |
| Tseng; Ping-sheng | Sunnyvale | CA | | |

US-CL-CURRENT: 703/16; 703/13, 703/17

ABSTRACT:

A method of simulating a system on a computer. The method comprises the following steps. First, analyze a hardware design language specification of the system to identify a set of processes. The hardware design language specification includes a register transfer level definition of a part of the system. Identify a set of triggered processes from the set of processes. Identify a set of triggers for the set of triggered processes, where a first trigger of the set of triggers is for causing a state change in a simulation of the system. Determine an evaluation order of the set of processes using the set of triggers. Simulate the system using the evaluation order.

19 Claims, 11 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 12

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

☐ 30. Document ID: US 5805792 A

L7: Entry 30 of 48 File: USPT Sep 8, 1998

US-PAT-NO: 5805792
DOCUMENT-IDENTIFIER: US 5805792 A

TITLE: Emulation devices, systems, and methods

DATE-ISSUED: September 8, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Ing-Simmons; Nicholas K. | Bedford | | | GB2 |
| Simpson; Richard David | Carlton | | | GB2 |

US-CL-CURRENT: 714/28; 714/25

ABSTRACT:

An electronic device having addressable storage elements and a bus so that the storage elements are accessible via the bus, an address register connected to the bus, a data register connected to the bus, terminals for serial scan-in and scan-out, a scanable emulation control register coupled to the terminals, and a selecting circuit responsive to bits in the emulation control register for coupling the address register and the data register to the terminals to enable scanning of the address and data registers.

7 Claims, 91 Drawing figures
Exemplary Claim Number: 2
Number of Drawing Sheets: 73



☑ 31. Document ID: US 5794046 A

L7: Entry 31 of 48 File: USPT Aug 11, 1998

US-PAT-NO: 5794046
DOCUMENT-IDENTIFIER: US 5794046 A

TITLE: Method and system for debugging parallel and distributed applications

DATE-ISSUED: August 11, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Meier; Michael S. | Newark | CA | | |
| Pan; Hsin | San Jose | CA | | |

US-CL-CURRENT: 717/128; 709/216, 714/45, 718/100, 719/320

ABSTRACT:

A debugger client/server application comprising a front-end and one or more back-ends, including a Director component which handles most of the initialization and parallel execution control issues and a rp.sub.-- client component and rp.sub.-- server component which handles most of the distributed execution issues. The Director allows a Debug Engine to be unaware of most of the parallel and distributed aspects of the application. Thus, the Debug Engine can be created by re-using a serial debugger for presenting the state information about the various programs that make up the application.

10 Claims, 4 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 3

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw. De

---

☐ 32. Document ID: US 5784593 A

L7: Entry 32 of 48 File: USPT Jul 21, 1998

US-PAT-NO: 5784593
DOCUMENT-IDENTIFIER: US 5784593 A
**** See image for Certificate of Correction ****

TITLE: Simulator including process levelization

DATE-ISSUED: July 21, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Tseng; Ping-sheng | Sunnyvale | CA | | |
| Vaidyanathan; Radha | Los Altos | CA | | |
| Nayudu; Sivaram Krishna | Jersey City | NJ | | |
| Ganapathi; Mahadevan | Sunnyvale | CA | | |

US-CL-CURRENT: 703/15

ABSTRACT:

A method of preparing a specification of a system for simulation on a computer system. The specification includes a hardware design language specification of the system. Analyze the specification to identify a set of processes, where each process includes a plurality of statements. Determine an evaluation order of the set of processes. Generate a combined process including a portion of the plurality of statements. The portion of the plurality of statements are included in the combined process so as to be evaluated according to the evaluation order.

16 Claims, 12 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 12

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw. De

---

☑ 33. Document ID: US 5784552 A

L7: Entry 33 of 48 File: USPT Jul 21, 1998

US-PAT-NO: 5784552
DOCUMENT-IDENTIFIER: US 5784552 A

TITLE: Debugging a computer program by simulating execution forwards and backwards in a main history log and alternative history logs

DATE-ISSUED: July 21, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Bishop; John E. | Nashua | NH | | |
| Carignan; Donald A. | Tyngsboro | MA | | |

US-CL-CURRENT: 714/38

ABSTRACT:

A computer program is executed in a forward direction to create a current state of registers and memory for the program. During the forward execution of the program, the pre-existing values of registers and memory changed by each instruction are recorded in a main log. During interactive debugging, reverse execution is simulated by displaying the contents of specified registers or memory locations. For each specified register or memory location, the main log is searched in a forward direction beginning at a specified time in the past and continuing until a value is found, or until the end of the main log is reached and a value is taken from the current state for the computer program. After simulated execution in reverse, the user may specify a changed value for a specified register or memory location, and then forward instruction interpretation is begun using the changed value, without changing the current state. New values of registers and memory locations generated by forward interpretation are recorded in an alternative log. Values of registers and memory accessed by forward-interpreted instructions are fetched by first searching the alternative log in a reverse direction, and when a value is not found in the alternative log, the main log is searched in a forward direction as described above. Moreover, at any time during forward interpretation, the user may specify a changed value, the change is logged in an alternative log, and forward interpretation continues.

22 Claims, 32 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 29



---

☑ 34. Document ID: US 5781778 A

L7: Entry 34 of 48 File: USPT Jul 14, 1998

US-PAT-NO: 5781778
DOCUMENT-IDENTIFIER: US 5781778 A

TITLE: Method and system for debugging parallel and distributed applications

DATE-ISSUED: July 14, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|

| | | |
|---|---|---|
| Meier; Michael S. | Newark | CA |
| Pan; Hsin | San Jose | CA |

US-CL-CURRENT: 717/127; 709/213, 709/216, 714/45, 718/100, 719/320

ABSTRACT:

A debugger client/server application comprising a front-end and one or more back-ends, including a Director component which handles most of the initialization and parallel execution control issues and a rp.sub.-- client component and rp.sub.-- server component which handles most of the distributed execution issues. The Director allows a Debug Engine to be unaware of most of the parallel and distributed aspects of the application. Thus, the Debug Engine can be created by re-using a serial debugger for presenting the state information about the various programs that make up the application.

5 Claims, 4 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 3



☐ 35. Document ID: US 5754759 A

L7: Entry 35 of 48 File: USPT May 19, 1998

US-PAT-NO: 5754759
DOCUMENT-IDENTIFIER: US 5754759 A

TITLE: Testing and monitoring of programmed devices

DATE-ISSUED: May 19, 1998

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Clarke; Paul A. | Crawley | | | GB2 |
| Piesing; Jonathan R. | Croydon | | | GB2 |

US-CL-CURRENT: 714/37; 714/31, 714/40

ABSTRACT:

A monitoring apparatus includes a programmable event filter (18) which identifies the occurrence of a predetermined signal (such as a memory address location) on a signal path of a programmed device such as a CD-i player. On detection, one or more successive bus signals are captured in a buffer (20) and subsequently written to storage (14) for processing by an internal processor (10) or the bus signals are sent over a bidirectional communications link (16) to a host device implementing suitable data processing algorithms. A particular use for the monitoring apparatus is in non-intrusive debugging of programmed devices taking account of their operating systems.

14 Claims, 14 Drawing figures

Exemplary Claim Number: 1
Number of Drawing Sheets: 7

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw De

---

☐ 36. Document ID: US 5687375 A

L7: Entry 36 of 48 File: USPT Nov 11, 1997

US-PAT-NO: 5687375
DOCUMENT-IDENTIFIER: US 5687375 A

TITLE: Debugging of High Performance Fortran programs with backup breakpoints

DATE-ISSUED: November 11, 1997

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Schwiegelshohn; Uwe | Dortmund | | | DE |

US-CL-CURRENT: 717/129; 714/35, 714/38, 717/149

ABSTRACT:

This invention is a debugger for HPF-like languages which can be implemented on top of basically any debugger. A primary feature of the debugger is the use of backup breakpoints to generate a program status which is similar to a program status in a sequential execution of the code and the back and forth mapping between processor variables. This debugger requires some new debugging information which must be provided by the compiler. It then allows debugging from a sequential point of view.

6 Claims, 6 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 4

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw De

---

☐ 37. Document ID: US 5675803 A

L7: Entry 37 of 48 File: USPT Oct 7, 1997

US-PAT-NO: 5675803
DOCUMENT-IDENTIFIER: US 5675803 A

TITLE: Method and apparatus for a fast debugger fix and continue operation

DATE-ISSUED: October 7, 1997

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|

| | | |
|---|---|---|
| Preisler; Thomas | Morgan Hill | CA |
| Gramlich; Wayne C. | Sunnyvale | CA |
| Pelegri-Llopart; Eduardo | Mountain View | CA |
| Miller; Terrence C. | Menlo Park | CA |

US-CL-CURRENT: 717/131

ABSTRACT:

This Continuation-In-Part describes a part of this run-time debugger operation which is designated the "Fix-and-Continue" invention. This invention permits a user to begin a debugging session wherein if an error in the code is encountered, the user can edit the corresponding source code to correct the error and then execute a "Fix and Continue" command all without leaving the debugging session. The Fix and Continue code calls the compiler to recompile the source code file with the edited text in it, receives the resulting recompiled object code file from the compiler, uses the dynamic linker to link the recompiled object code into the target application program process, patches the previous version of this same object code fie to refer to the newly recompiled code, resets any required variables and registers, resets the program counter to the line of code being executed when the error was discovered. The debugger then continues in the debug session thereby saving the time it would ordinarily take to quit the debug session, relink and reload the target program and start the debug session once again.

20 Claims, 11 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 10



---

☐ 38. Document ID: US 5630049 A

L7: Entry 38 of 48 File: USPT May 13, 1997

US-PAT-NO: 5630049
DOCUMENT-IDENTIFIER: US 5630049 A

TITLE: Method and apparatus for testing software on a computer network

DATE-ISSUED: May 13, 1997

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Cardoza; Wayne M. | Amherst | NH | | |
| Diewald; Jeffrey M. | Billerica | MA | | |
| Nelson; Jeffrey E. | Milford | NH | | |
| DiPirro; Steven D. | Amherst | NH | | |
| Goddard; James R. | Candia | NH | | |
| Fisher, Jr.; Wendell B. | Nashua | NH | | |
| McElearney; Anne E. | Groton | MA | | |
| Sayde; Richard | Littleton | MA | | |

US-CL-CURRENT: 714/25

ABSTRACT:

A method of remote debugging comprises a first computer system that communicates with a second computer using a network connection. The first computer system controls the remote debugging and comprises a first operating system. The second computer system comprises a second operating system and software being tested. User input, in the form of debug commands, is received using a remote debugger in the first computer system to control the remote debugging session. The remote debugger translates a debug command into messages that are sent from the first computer system to the second computer system. The messages correspond to tasks that the target computer system performs to complete the debug command. During debugging, the target computer system transitions between polling or stopped mode and interrupt-driven mode by transitioning both the target operating system and network hardware in the target computer system that interfaces with the network.

22 Claims, 11 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 10



☐ 39. Document ID: US 5621651 A

L7: Entry 39 of 48 File: USPT Apr 15, 1997

US-PAT-NO: 5621651
DOCUMENT-IDENTIFIER: US 5621651 A

TITLE: Emulation devices, systems and methods with distributed control of test interfaces in clock domains

DATE-ISSUED: April 15, 1997

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |

US-CL-CURRENT: 703/23; 703/13

ABSTRACT:

An emulation device (11) distributes common control information (8801) to each of a plurality of clock domains (1213, 1215, 1217) into which the emulation device is partitioned, and also provides the clock domains with individualized clock control (8905, 8907, 8913).

12 Claims, 94 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 75

☑ 40. Document ID: US 5583988 A

L7: Entry 40 of 48 File: USPT Dec 10, 1996

US-PAT-NO: 5583988
DOCUMENT-IDENTIFIER: US 5583988 A
**** See image for Certificate of Correction ****

TITLE: Method and apparatus for providing runtime checking features in a compiled programming development environment

DATE-ISSUED: December 10, 1996

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Crank; Erik | Travis County | TX | | |
| Bellin; Jon | Travis County | TX | | |

US-CL-CURRENT: 714/48; 714/38

ABSTRACT:

A method and apparatus for performing runtime checking during program execution in a compiled environment using the full ANSI-C programming language. The present invention detects a number of errors during runtime that cannot be found by a compiler at the precise moment that a respective C language restriction is violated. The present invention also provides the user with a direct indication of the problem, thus saving debugging time. The runtime checking features of the present invention further detects when a user is using library functions improperly. When C source code is compiled, the present invention allocates special data structures for every pointer, array and structure object in the program. An association is made between each of these objects, and its special data structure in the compiler symbol table. At runtime, these data structures contain status information about their associated objects. The present invention also inserts special machine language instructions for C expressions during compilation that either modify values in the special data structures or call internal runtime checking functions according to the present invention that use the information in the respective data structures to determine whether an expression is illegal and report errors if necessary. The runtime checking features of the present invention include a method for specifying precise restrictions on the arguments that may be passed to library functions. These restrictions are used to determine whether arguments to library functions conform to their respective restrictions and reports any violations to the user at runtime, indicating which argument caused the error and which restriction was violated.

45 Claims, 44 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 41

☐ 41. Document ID: US 5535331 A

L7: Entry 41 of 48 File: USPT Jul 9, 1996

US-PAT-NO: 5535331
DOCUMENT-IDENTIFIER: US 5535331 A

TITLE: Processor condition sensing circuits, systems and methods

DATE-ISSUED: July 9, 1996

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Ehlig; Peter N. | Houston | TX | | |

US-CL-CURRENT: 714/45; 714/28, 714/30

ABSTRACT:

Operations of a data processing device are traced by detecting a jump address in the program counter sequence, and pushing the jump address onto a trace stack.

16 Claims, 91 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 73



☐ 42. Document ID: US 5394544 A

L7: Entry 42 of 48 File: USPT Feb 28, 1995

US-PAT-NO: 5394544
DOCUMENT-IDENTIFIER: US 5394544 A

TITLE: Software system debugger with distinct interrupt vector maps for debugging and application programs

DATE-ISSUED: February 28, 1995

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Motoyama; Tetsuro | San Jose | CA | | |
| Mor; Banky | San Jose | CA | | |
| Rodriquez; Gregorio | Union City | CA | | |
| Kim; Chan | Fremont | CA | | |

US-CL-CURRENT: 714/31; 714/38

ABSTRACT:

A system for debugging an application program includes a debugging execution unit operated under an operating-system-free environment. The debugging execution unit includes a communication channel, a target processor unit for executing the application program and a first debugging program, and an interrupt vector swapper. The first debugging program includes a first set of input/output procedures for handling debugging communications over the communication channel, and the application program includes a second set of input/output procedures for handling communications over the communication channel associated with execution of the application program. Interrupt signals are generated when the target processor unit receives communications from the host processor unit. The target processor unit communicates with a host processor unit via the communication channel both when executing the application program and when executing the first debugging program. A first set of interrupt vector entries link interrupt signals generated during execution of the first debugging program to a first set of destination addresses. A second set of interrupt vector entries link interrupt signals generated during execution of the application program to a second set of destination addresses. A vector controller maps interrupt signals via the first set of interrupt vector entries during execution of the first debugging program, and maps interrupt signals via the second set of interrupt vector entries during execution of the application program.

13 Claims, 13 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 12



---

☐ 43. Document ID: US 5329471 A

L7: Entry 43 of 48 File: USPT Jul 12, 1994

US-PAT-NO: 5329471
DOCUMENT-IDENTIFIER: US 5329471 A
**** See image for Certificate of Correction ****

TITLE: Emulation devices, systems and methods utilizing state machines

DATE-ISSUED: July 12, 1994

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Swoboda; Gary L. | Sugar Land | TX | | |
| Daniels; Martin D. | Houston | TX | | |
| Coomes; Joseph A. | Missouri City | TX | | |

US-CL-CURRENT: 703/23; 703/13, 714/28, 714/30, 714/727

ABSTRACT:

An emulation device including a serial scan testability interface having at least first and second scan paths, and state machine circuitry connected and responsive to said second scan path generally operable for emulation control of logical

circuitry associated with said emulation device.

69 Claims, 47 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 41

Full | Title | Citation | Front | Review | Classification | Date | Reference | [illegible] | [illegible] | Claims | KWIC | Draw. D

---

☐ 44. Document ID: US 5321828 A

L7: Entry 44 of 48 File: USPT Jun 14, 1994

US-PAT-NO: 5321828
DOCUMENT-IDENTIFIER: US 5321828 A

TITLE: High speed microcomputer in-circuit emulator

DATE-ISSUED: June 14, 1994

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Phillips; Michael D. | Santa Clara | CA | | |
| Wilburn; Darrell L. | Saratoga | CA | | |
| Hua; Van T. | San Jose | CA | | |
| Minami; Gordon A. | Palo Alto | CA | | |
| Kresge; Robert J. | Cupertino | CA | | |
| Verhaegh; Charles | Arkata | CA | | |

US-CL-CURRENT: 703/28

ABSTRACT:

An in-circuit emulator (ICE) for hardware/software development and debugging microprocessors. Program execution reconstruction is extracted from an on-board cache memory. An external ICE enclosure interfaces to a target system microprocessor via a cable and a buffer/interface pod. A control program directs a non-intrusive emulation and a monitor program resides in a personal computer host and supports ICE commands. The monitor program allows a user to follow a target system's program flow, to capture related processor information, to make program modifications, and allows the user to restart programs. An on-line disassembler presents a display so as to allow the designer to examine memory, using instruction mnemonics rather than hexadecimal values, thus improving the designer's ability to read program memory. A bit trace buffer records the state of each the microprocessor's signals during each cycle of each instruction. Multiple breakpoints allow a system developer to control a program in ROM, as well as one resident in RAM. An external-range hardware breakpoint and up to sixteen software breakpoints are provided and these allow a designer to display, set and reset breakpoint addresses.

9 Claims, 15 Drawing figures
Exemplary Claim Number: 7
Number of Drawing Sheets: 7

☐ 45. Document ID: US 5175856 A

L7: Entry 45 of 48 File: USPT Dec 29, 1992

US-PAT-NO: 5175856
DOCUMENT-IDENTIFIER: US 5175856 A
**** See image for Certificate of Correction ****

TITLE: Computer with integrated hierarchical representation (IHR) of program wherein IHR file is available for debugging and optimizing during target execution

DATE-ISSUED: December 29, 1992

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Van Dyke; Don A. | Pleasanton | CA | | |
| Cramer; Timothy J. | Pleasanton | CA | | |
| Rasbold; James C. | Livermore | CA | | |
| O'Hair; Kelly T. | Livermore | CA | | |
| Cox; David M. | Livermore | CA | | |
| Seberger; David A. | Livermore | CA | | |
| O'Gara; Linda J. | Livermore | CA | | |
| Masamitsu; Jon A. | Livermore | CA | | |
| Strout, II; Robert E. | Livermore | CA | | |
| Chandramouli; Ashok | Fremont | CA | | |

US-CL-CURRENT: 717/151; 717/124, 717/159

ABSTRACT:

A modular compilation system that utilizes a fully integrated hierarchical representation as a common intermediate representation to compile source code programs written in one or more procedural programming languages into an executable object code file. The structure of the integrated common intermediate representation supports machine-independent optimizations, as well as machine-dependent optimizations, and also supports source-level debugging of the executable object code file. The integrated hierarchical representation (IHR) is language independent and is shared by all of the components of the software development system, including the debugger.

18 Claims, 16 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 16

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWMC | Draw. D

☐ 46. Document ID: US 5127103 A

L7: Entry 46 of 48 File: USPT Jun 30, 1992

US-PAT-NO: 5127103
DOCUMENT-IDENTIFIER: US 5127103 A

TITLE: Real-time tracing of dynamic local data in high level languages in the presence of process context switches

DATE-ISSUED: June 30, 1992

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Hill; Charles R. | Peekskill | NY | | |
| Tyra; Fryderyk | Yonkers | NY | | |
| Akiwumi-Assani; Samuel O. | Beacon | NY | | |

US-CL-CURRENT: 714/45

ABSTRACT:

An improved real-time debugger accommodates high level language computer programs containing dynamic local data and process context switches. Information thus acquired is used to deduce the stack frame pointer. Inputs and outputs of a target processor are tapped to capture key instructions, particularly indicating context switches. A local tag memory in the debugger stores images of stack frames during context switches.

30 Claims, 20 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 16



☑ 47. Document ID: US 5093914 A

L7: Entry 47 of 48 File: USPT Mar 3, 1992

US-PAT-NO: 5093914
DOCUMENT-IDENTIFIER: US 5093914 A

TITLE: Method of controlling the execution of object-oriented programs

DATE-ISSUED: March 3, 1992

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Coplien; James O. | Wheaton | IL | | |
| Williams; Thomas V. | Naperville | IL | | |

US-CL-CURRENT: 717/129

ABSTRACT:

A method used by a digital computer in controlling execution of an object-oriented program to effect a defined action, e.g., stopping the program, when a specified virtual function is invoked on a specified object during execution of the program. A breakpoint address is determined at run time, advantageously after the specified object is created in accordance with execution of the program. The breakpoint address determination is not based solely on symbol table, pre-execution, information, but in addition on information generated in conjunction with the creation of the specified object. The breakpoint is inserted while program execution is stopped at an intermediate program point after the specified object is created. After program execution is resumed and the specified virtual function is invoked in accordance with the program, the breakpoint fires. However, the defined action is performed only in response to determining that the firing occurred on the specified object.

21 Claims, 11 Drawing figures
Exemplary Claim Number: 14
Number of Drawing Sheets: 11



☐ 48. Document ID: US 4558413 A

L7: Entry 48 of 48 File: USPT Dec 10, 1985

US-PAT-NO: 4558413
DOCUMENT-IDENTIFIER: US 4558413 A

TITLE: Software version management system

DATE-ISSUED: December 10, 1985

INVENTOR-INFORMATION:

| NAME | CITY | STATE | ZIP CODE | COUNTRY |
|---|---|---|---|---|
| Schmidt; Eric E. | Los Altos | CA | | |
| Lampson; Butler W. | Philadelphia | PA | | |

US-CL-CURRENT: 707/203; 717/110, 717/145, 717/171

ABSTRACT:

A software version management system, also called system modeller, provides for automatically collecting and recompiling updated versions of component software objects comprising a software program for operation on a plurality of personal computers coupled together in a distributed software environment via a local area network. The component software objects include the source and binary files for the software program, which stored in various different local and remote storage means through the environment. The component software objects are periodically updated, via a system editor, by various users at their personal computers and then stored in designated storage means. The management system includes models which are also objects. Each of the models is representative of the source versions of a particular component software object and contain object pointers including a unique name of the object, a unique identifier descriptive of the cronological updating of

its current version, information as to an object's dependencies on other objects and a pathname representative of the residence storage means of the object. Means are provided in the system editor to notify the management system when any one of the objects is being edited by a user and the management system is responsive to such notification to track the edited objects and alter their respective models to the current version thereof.

6 Claims, 29 Drawing figures
Exemplary Claim Number: 1
Number of Drawing Sheets: 24



Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

| Terms | Documents |
|---|---|
| L6 AND (symbol ADJ table) | 48 |

**Display Format:** REV Change Format

Previous Page Next Page Go to Doc#

Titles of Most Frequently Occurring Classifications of Patents Returned

From A Search of 09864109 on April 21, 2004

15 714/38 (10 OR, 5 XR)
Class 714 : ERROR DETECTION/CORRECTION AND FAULT DETECTION/RECOVERY
714/100 DATA PROCESSING SYSTEM ERROR OR FAULT HANDLING

714/1 .Reliability and availability
714/25 ..Fault locating (i.e., diagnosis or testing)

714/37 ...Analysis (e.g., of output, state, or design)

714/38 ....Of computer software

7 712/227 (3 OR, 4 XR)
Class 712 : ELECTRICAL COMPUTERS AND DIGITAL PROCESSING SYSTEMS: PROCESSING ARCHITECTURES AND INSTRUCTION PROCESSING
712/220 PROCESSING CONTROL
712/227 .Specialized instruction processing in support of testing, debugging, emulation

7 714/35 (1 OR, 6 XR)
Class 714 : ERROR DETECTION/CORRECTION AND FAULT DETECTION/RECOVERY
714/100 DATA PROCESSING SYSTEM ERROR OR FAULT HANDLING

714/1 .Reliability and availability
714/25 ..Fault locating (i.e., diagnosis or testing)

714/32 ...Particular stimulus creation
714/35 ....Substituted or added instruction (e.g., code instrumenting, breakpoint instruction)

5 714/45 (1 OR, 4 XR)
Class 714 : ERROR DETECTION/CORRECTION AND FAULT DETECTION/RECOVERY
714/100 DATA PROCESSING SYSTEM ERROR OR FAULT HANDLING

714/1 .Reliability and availability
714/25 ..Fault locating (i.e., diagnosis or testing)



search strategy

714/45 ...Output recording (e.g., signature or trace)

5 717/124 (3 OR, 2 XR)
Class 717 : DATA PROCESSING: SOFTWARE DEVELOPMENT, INSTALLATION, AND MANAGEMENT
717/100 SOFTWARE PROGRAM DEVELOPMENT TOOL (E.G., INTEGRATED CASE TOOL OR STAND-ALONE DEVELOPMENT TOOL)
717/124 .Testing or debugging

4 714/34 (0 OR, 4 XR)
Class 714 : ERROR DETECTION/CORRECTION AND FAULT DETECTION/RECOVERY
714/100 DATA PROCESSING SYSTEM ERROR OR FAULT HANDLING

714/1 .Reliability and availability
714/25 ..Fault locating (i.e., diagnosis or testing)

714/32 ...Particular stimulus creation
714/34 ....Halt, clock, or interrupt signal (e.g., freezing, hardware breakpoint, single-stepping)

4 717/125 (2 OR, 2 XR)
Class 717 : DATA PROCESSING: SOFTWARE DEVELOPMENT, INSTALLATION, AND MANAGEMENT
717/100 SOFTWARE PROGRAM DEVELOPMENT TOOL (E.G., INTEGRATED CASE TOOL OR STAND-ALONE DEVELOPMENT TOOL)
717/124 .Testing or debugging
717/125 ..Having interactive or visual

4 717/127 (3 OR, 1 XR)
Class 717 : DATA PROCESSING: SOFTWARE DEVELOPMENT, INSTALLATION, AND MANAGEMENT
717/100 SOFTWARE PROGRAM DEVELOPMENT TOOL (E.G., INTEGRATED CASE TOOL OR STAND-ALONE DEVELOPMENT TOOL)
717/124 .Testing or debugging
717/127 ..Monitoring program execution

3 717/129 (2 OR, 1 XR)
Class 717 : DATA PROCESSING: SOFTWARE DEVELOPMENT, INSTALLATION, AND MANAGEMENT
717/100 SOFTWARE PROGRAM DEVELOPMENT TOOL (E.G., INTEGRATED CASE TOOL OR STAND-ALONE DEVELOPMENT TOOL)

717/124 .Testing or debugging
717/127 ..Monitoring program execution
717/129 ...Using breakpoint

2 345/967 (0 OR, 2 XR)
Class 345 : COMPUTER GRAPHICS PROCESSING, OPERATOR INTERFACE PROCESSING, AND SELECTIVE VISUAL DISPLAY SYSTEMS
345/961 OPERATOR INTERFACE WITH VISUAL STRUCTURE OR FUNCTION DICTATED BY INTENDED USE
345/965 .For process control and configuration
345/966 ..Computer process (e.g., operation of computer)
345/967 ...Visual or iconic programming

2 703/22 (1 OR, 1 XR)
Class 703 : DATA PROCESSING: STRUCTURAL DESIGN, MODELING, SIMULATION, AND EMULATION
703/13 SIMULATING ELECTRONIC DEVICE OR ELECTRICAL SYSTEM
703/22 .Software program (i.e., performance prediction)

2 703/28 (1 OR, 1 XR)
Class 703 : DATA PROCESSING: STRUCTURAL DESIGN, MODELING, SIMULATION, AND EMULATION
703/23 EMULATION
703/28 .In-circuit emulator (i.e., ICE)

2 709/203 (0 OR, 2 XR)
Class 709 : ELECTRICAL COMPUTERS AND DIGITAL PROCESSING SYSTEMS: MULTIPLE COMPUTER OR PROCESS COORDINATING
709/200 MULTICOMPUTER DATA TRANSFERRING
709/201 .Distributed data processing
709/203 ..Client/server

2 709/208 (0 OR, 2 XR)
Class 709 : ELECTRICAL COMPUTERS AND DIGITAL PROCESSING SYSTEMS: MULTIPLE COMPUTER OR PROCESS COORDINATING
709/200 MULTICOMPUTER DATA TRANSFERRING
709/208 .Master/slave computer controlling

2 714/28 (1 OR, 1 XR)
Class 714 : ERROR DETECTION/CORRECTION AND FAULT DETECTION/RECOVERY